API文档-云盾身份认证（实名二要素）

一. 产品说明

接入问题及获取无授权模式请加钉钉群：21727262

二. 产品功能

1. 功能说明：通过验证用户身份证号和姓名是否一致，来确定用户身份是否虚假，达到用户合规性要求。
2. 使用流程：

- 步骤一：购买产品，见本页“立即购买”
- 步骤二：产品接入，见本页“请求示例”

三. 产品接入说明

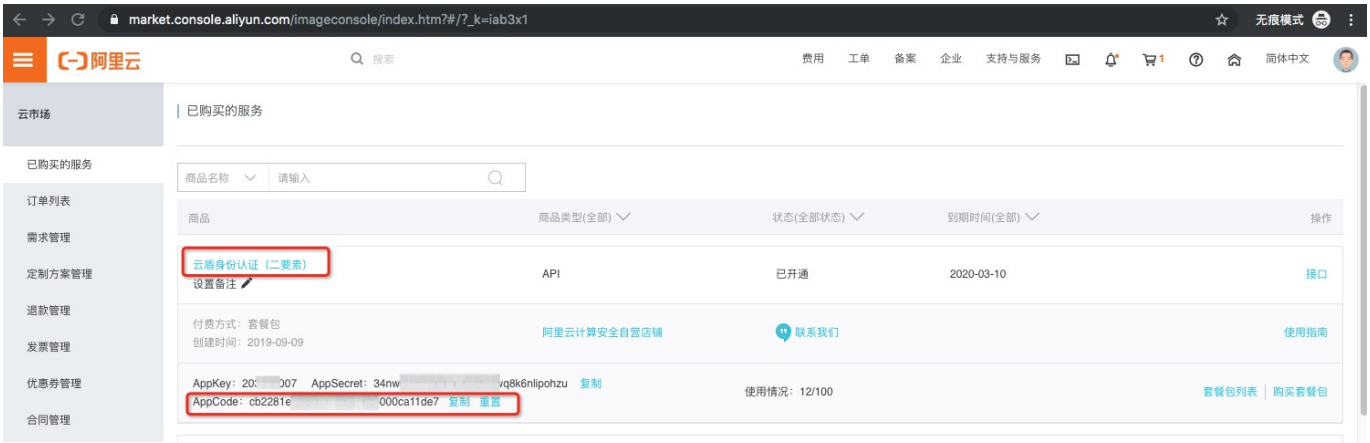
1. 入参说明

名称	类型	是否必选	说明
appcode	STRING	必选	调用产品的身份认证，查看appcode值：https://market.console.aliyun.com/imageconsole/index.htm?#/bizlist
verifyKey	STRING	必选	调用产品的身份认证，通过控制台创建verifyKey（必须由主账号创建）：https://yundunnext.console.aliyun.com/?p=saf#/config
__userId	STRING	必选	购买产品的主账号ID，查看主账号ID：https://account.console.aliyun.com/#/secure
customerID	STRING	可选	用户账号ID，不影响实时验证结果，可用于问题排查和定位
identifyNum	STRING	可选	需验证用户的身份证
userName	STRING	可选	需验证用户的姓名

2. 入参获取

2.1. 查看appcode

- 购买产品之后才有appcode
- https://market.console.aliyun.com/imageconsole/index.htm?#/bizlist



2.2. 查看主账号ID

- 需要主账号登录阿里云官网
- 地址：https://account.console.aliyun.com/#/secure

2.3. 创建verifyKey

- VerifyKey必须由阿里云主账号创建
- 地址：https://yundunnext.console.aliyun.com/?p=saf#/config

1. 点击“新增VerifyKey”：

2、填入域名（公司域名），选择操作端，然后点击确认

3、页面返回VerifyKey信息

3. 产品调用模式

身份证、姓名、手机号三要素涉及用户隐私数据，本产品提供两种用户授权方式：

1）免授权模式

加钉钉群（21727262）找客服，咨询身份验证客户获取用户授权协议，加入自己业务的用户协议中，把用户协议地址给到客服确认后，客服配置免授权模式，接口直接返回认证结果信息。

2）授权模式

按照请求示例接入，调通API后返回授权链接，在前端打开URL后用户点击授权后可以通过解签获取验证结果信息。具体验证流程见文档(入参获取和授权说明)。

四. 产品接入Demo

1. Java POM依赖：

```
<dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>fastjson</artifactId>
    <version>1.2.60</version>
</dependency>
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>4.2.1</version>
</dependency>
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpcore</artifactId>
    <version>4.2.1</version>
```

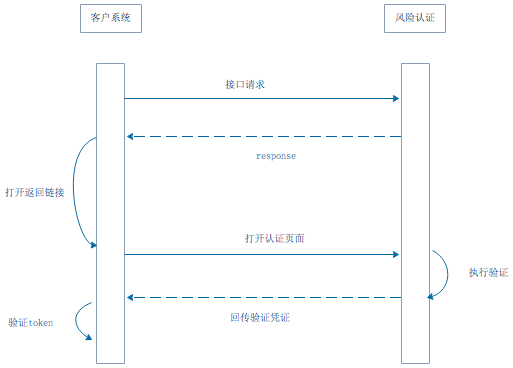
```
</dependency>
<dependency>
    <groupId>commons-lang</groupId>
    <artifactId>commons-lang</artifactId>
    <version>2.6</version>
</dependency>
```

2. Java代码Demo

```
import java.util.HashMap;
import java.util.Map;

import org.apache.http.HttpResponse;
import org.apache.http.util.EntityUtils;

public class Safrv2metaIdName {

    public static void main(String[] args) {
        String host = "https://safrvcert.market.alicloudapi.com";
        String path = "/safrv_2meta_id_name/";
        String method = "GET";
        // 查看链接：https://market.console.aliyun.com/imageconsole/index.htm?#/bizlist?_k=cgs1jr
        String appcode = "自己的appcode";
        Map<String, String> headers = new HashMap<String, String>();
        //最后在header中的格式(中间是英文空格)为Authorization:APPCODE 83359fd73fe94948385f570e3c139105
        headers.put("Authorization", "APPCODE " + appcode);
        Map<String, String> querys = new HashMap<String, String>();
        querys.put("__userId", "主账号ID");
        querys.put("customerID", "12345678");
        querys.put("identifyNum", "身份证号");
        querys.put("userName", "姓名");
        //创建和查询链接：https://yundunnext.console.aliyun.com/?p=saf#/config
        querys.put("verifyKey", "自己的verifyKey");

```

```
        try {
            /**
             * 重要提示如下:
             * HttpUtils请从
             * https://github.com/aliyun/api-gateway-demo-sign-java/blob/master/src/main/java/com/aliyun/api/gateway/demo/util/HttpUtils.java
             */
            HttpResponse response = HttpUtils.doGet(host, path, method, headers, querys);
            System.out.println(response.toString());

            //查看云市场报错信息
            //for (Header header : response.getAllHeaders()) {
            //    if(header.getName().contains("X-Ca-Error-Message")) {
            //        System.out.println("Error message: " + header.getValue());
            //    }
            //}

            //获取response的body
            System.out.println(EntityUtils.toString(response.getEntity()));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

3. 授权模式验签



用户在认证页面，身份验证成功之后，验证页面会把验证结果token回传给接入应用，这个验证结果token是标识当次身份验证是否有效的关键，防止恶意篡改，接入应用必须要对这个token进行签名验证。

- 身份验证页面回传token给接入应用的两种方式：

1）方式一：callBack回到接入的应用系统

比如在无线端接入身份验证的H5页面时，身份验证成功会跳回到接入应用的业务页面，并将token和签名信息放在url中带给接入应用。

- 获取token及签名的Demo：

```
String ivSign = request.getParameter("ivSign");
//进行验签操作...
```

2）方式二：仅支持mini框的接入方式，不需要传入callBack参数

通过javascript的postMessage机制发送消息告诉接入应用的业务页面。比如在PC上通过MINI弹窗来接入身份验证的页面时，身份验证成功会将成功的token和签名postMessage会接入应用的业务页面。

- 获取token及签名的Demo：

```
window.addEventListener("message", function( event ) {
    var json = JSON.parse(decodeURIComponent(event.data));
    if('type' in json && json.type=='iframevalid'){
        var ivSign = json.ivSign;
        //进行验签操作...
    }
} );
```

对验证成功token进行验签，（通过前面申请的VerifySecret进行验签）

- 示例Java Demo：

```
/**
 * 参数均是从请求中获取
 * @param ivSign    验证成功之后的回传参数sign
 * @param verifySecret  在接入时分配的verifySecret
 */
   public static void verifySign(String ivSign,String verifySecret)
{
        try{
            byte[] encryptedData = java.util.Base64.getDecoder().dec
ode(ivSign.getBytes("utf-8"));
            byte[] keyData = java.util.Base64.getDecoder().decode(ve
rifySecret.getBytes());
            PKCS8EncodedKeySpec keySpec = new PKCS8EncodedKeySpec(ke
yData);
            KeyFactory keyFactory = KeyFactory.getInstance("RSA");
            RSAPrivateKey privateKey = (RSAPrivateKey) keyFactory.ge
neratePrivate(keySpec);
            Cipher cipher = Cipher.getInstance("RSA");
            cipher.init(Cipher.DECRYPT_MODE, privateKey);
            String content = new String(doBlock(encryptedData, priva
teKey.getModulus().bitLength()/8, cipher),"utf-8");
            Map<String,String> map = JSON.parseObject(content,Map.cl
ass);
            //校验通过
            if(map!=null && map.containsKey("success")){
                boolean valid = false;
                String ivTs = map.get("signTs");
                if(ivTs!=null && !"".equals(ivTs)){
                    Long expiredTime = Long.parseLong(ivTs)+300;
                    if(expiredTime>System.currentTimeMillis()/1000){
                        valid = true;
                    }
                }
                //签名未过期
                if(valid){
                    if(Boolean.valueOf(map.get("success"))) {
                        //认证通过，继续业务操作
                        String customerId = map.get("customerId");
```

```
                    }else{
                        //认证不通过
                    }
                }
            }else{
                //签名无效 阻断
                return;
            }
        }catch (Exception e){

        }
    }

    private static byte[] dolock(byte[] encryptedData, int blockSize, Cipher cipher) throws IllegalBlockSizeException, BadPaddingException {
        int inputLen = encryptedData.length;
        ByteArrayOutputStream out = new ByteArrayOutputStream();
        int offSet = 0;
        byte[] cache;
        int i = 0;
        // 对数据分段解密
        while (inputLen - offSet > 0) {
            if (inputLen - offSet > blockSize) {
                cache = cipher.doFinal(encryptedData, offSet, blockSize);
            } else {
                cache = cipher.doFinal(encryptedData, offSet, inputLen - offSet);
            }
            out.write(cache, 0, cache.length);
            i++;
            offSet = i * blockSize;
        }

        byte[] decryptedData = out.toByteArray();
        IOUtils.closeQuietly(out);
        return decryptedData;
    }
```

- 示例PHP Demo：

```php
function verifySign($ivSign,$verifySecret) {
        $content="";
        $verifySecret = chunk_split($verifySecret,64,"\n");
        $verifySecret = "-----BEGIN RSA PRIVATE KEY-----\n".$verifySecret."-----END RSA PRIVATE KEY-----\n";
        $pri_key = openssl_get_privatekey($verifySecret);
        $key_len = openssl_pkey_get_details($pri_key)['bits'];
        $part_len = $key_len / 8;
        $encrypted = base64_decode(rawurldecode($ivSign));
        $parts = str_split($encrypted, $part_len);
        foreach ($parts as $part) {
            $decrypted_temp = '';
            openssl_private_decrypt($part, $decrypted_temp,$pri_key);
            $content .= $decrypted_temp;
        }
        $content = iconv("UTF-8",iconv_get_encoding("internal_encoding"),$content);
        $json = json_decode($content);
        if(intval($json->{'signTs'})+300>time()){
                  //签名未过期，继续业务操作
           if($json->{'success'}=='true'){
               //标识成功了
           }else{
               //标识失败
           }
        }else{
          echo "expired";
        }
}
```

五. 出参说明

- 免授权模式返回值示例

```
{
    "code":200,
    "value":{
        "bizCode":0,
        "message":"success"
    },
    "message":"success"
}
```

- 授权模式返回值示例

```
{
    "code": 200,
    "value": {
        "verifyUrl": "https://yuniv.aliyuncs.com/iv/remote/h5/request.htm?havana_iv_token=AQgAENgEIg1oYXZhbmFfYXBwX2l2MgEBOPb97pPKLEABShBVBpCDElXO6cLLSyS6gnySaQyCdDcggxpqTC1u0d1AdnEfX-A"
    },
    "message": "success"
}
```

- 返回值说明

名称	类型	是否必选	说明
code	STRING	必选	调用码，具体见【调用码说明】
value.verifyUrl	STRING	可选	用户验证链接，授权流程见下文【授权说明】
value.bizCode	STRING	可选	业务码，具体见【业务码说明】
message	STRING	必选	返回信息说明

- 调用码code说明

code值	说明
200	调用成功
402	调用错误
403	无权限调用
500	内部服务错误

501	无此记录

- 调用码code=200时，业务码bizCode说明

bizCode值	说明
0	验证通过
13066	验证不一致